Tworzenie Aplikacji Mobilnych

Dokumentacja projektowa

Kamil Ćwikowski 108610

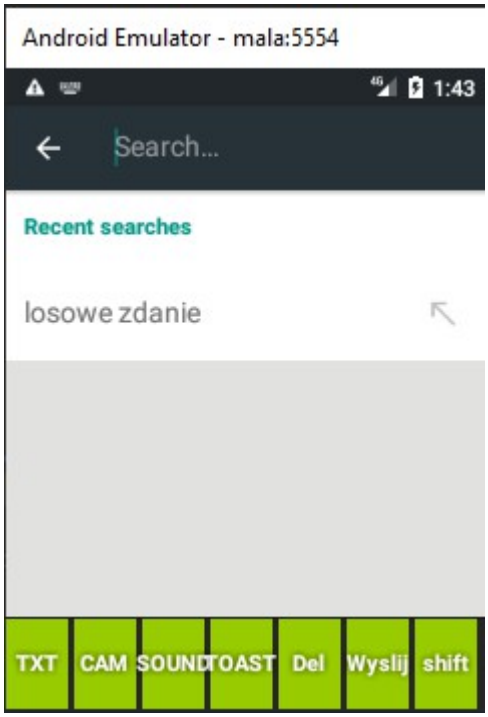
Teleinformatyka 4 rok (semestr VII)

Uniwersytet Technologiczno-Przyrodniczy im. Jana i Jędrzeja Śniadeckich
Wydział Telekomunikacji, Informatyki i Elektrotechniki

Bydgoszcz, 27.01.2020

Projekt zakładał utworzenie aplikacji mobilnej na platformę Android, która będzie działa jako klawiatura systemowa systemowa spełniająca następujące funkcje

- Wprowadzenie tekstu np. '' I'm custom Keyboard!”
- Odtworzenie dźwięku
- Zapis do pliku dowolnego tekstu
- Wyświetlenie Toast z tekstem
- Włączenie aparatu
- Zmiana położenia klawiszy/ rozszerzenie o nowe akcje
- Sprawdzenie, czy moduł NFC jest włączony
- Włączenie/Wyłączenie modułu NFC
- Oraz dwie inne własne funkcjonalności



W ramach wyboru własnych funkcjonalności, utworzona w projecie klawiatura usuwa pojedyncze znaki z pola tekstowego, oraz sprawdza, czy adapter BT jest włączony.

W ramach Kolejnego etapu projektu, klawiatura została rozszerzona o komunikację Bluetooth, oraz przesyłanie następujących komunikatów:

- Hello I'm Custom Keyboard!
- 6000010000123
- 6000010000124

Poszczególne linie kodu aplikacji

```xml
<?xml version="1.0" encoding="utf-8"?>
<Keyboard xmlns:android="http://schemas.android.com/apk/res/android"
    android:keyWidth="16.6%p"
    android:horizontalGap="5dp"
    android:verticalGap="5dp"
    android:keyHeight="60dp">


    <Row>
        <Key android:codes= "1" android:keyLabel="TXT" android:keyEdgeFlags="left"/>
        <Key android:codes= "61" android:keyLabel="CAM"/>
        <Key android:codes="3" android:keyLabel="SOUND"/>
        <Key android:codes="4" android:keyLabel="TOAST"/>
        <Key android:codes="-5" android:keyLabel="Del"/>
        <Key android:codes="77" android:keyLabel="Wyslij"/>
        <Key android:codes="15" android:keyLabel="shift" android:keyEdgeFlags="right"/>
    </Row>




</Keyboard>
```

Ilustracja 1: Klawiatura

```java
public class MyInputMethodService extends InputMethodService implements KeyboardView.OnKeyboardActionListener {
    @Override
    public View onCreateInputView() {
        KeyboardView keyboardView = (KeyboardView) getLayoutInflater().inflate(R.layout.keyboard_view, root: null);
        Keyboard keyboard = new Keyboard( context: this, R.layout.number_pad);
        keyboardView.setKeyboard(keyboard);
        keyboardView.setOnKeyboardActionListener(this);
        return keyboardView;
    }

    public View onCreateInputView2() {
        KeyboardView keyboardView = (KeyboardView) getLayoutInflater().inflate(R.layout.keyboard_view, root: null);
        Keyboard keyboard = new Keyboard( context: this, R.layout.number_pad2);
        keyboardView.setKeyboard(keyboard);
        keyboardView.setOnKeyboardActionListener(this);
        return keyboardView;
    }
    public View onCreateInputView3() {
        KeyboardView keyboardView = (KeyboardView) getLayoutInflater().inflate(R.layout.keyboard_view, root: null);
        Keyboard keyboard = new Keyboard( context: this, R.layout.number_pad3);
        keyboardView.setKeyboard(keyboard);
        keyboardView.setOnKeyboardActionListener(this);
        return keyboardView;
    }
```

Ilustracja 2: Wywołanie widoków

```
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.CAMERA"> </uses-permission>
<uses-feature android:name="android.hardware.camera2" />
<uses-feature android:name="android.hardware.nfc"  android:required="false" />
<uses-permission android:name="android.permission.NFC" />
```

Ilustracja 3: Pozwolenia dla poszczególnych intencji

```
public void serwer(String tekst)
{
    final BluetoothServerSocket mmServerSocket;
    BluetoothAdapter mBluetoothAdapter = BluetoothAdapter.getDefaultAdapter();

    BluetoothServerSocket tmp = null;
    try {
        UUID uuid=UUID.fromString("ebe3a525-0cef-4907-94b0-e39e1255ce43");
        tmp = mBluetoothAdapter.listenUsingRfcommWithServiceRecord( name: "Usługa witająca", uuid);
    } catch (IOException e) { }
    mmServerSocket = tmp;

    BluetoothSocket socket;
    while (true) {
        Log.d( tag: "INFO", msg: "czas");
        try {
            Log.d( tag: "INFO", msg: "Czekam na połączenie od clienta");
            socket = mmServerSocket.accept();
            Log.d( tag: "INFO", msg: "Mam clienta!");
            String text = "TOAST!";
            Toast.makeText(getApplicationContext(), text,  duration: 5).show();
            PrintWriter out = new PrintWriter(socket.getOutputStream(), autoFlush: true);
            out.println(tekst);
        } catch (IOException e) {
            break;

        }
        if (socket != null) {

            try {
                mmServerSocket.close();


            } catch (Exception e) {
```

Ilustracja 4: Komunikacja Bluetooth po stronie serwera

```
public void klient()
{
    BluetoothAdapter bal = BluetoothAdapter.getDefaultAdapter();

    BluetoothDevice serwer = bal.getRemoteDevice("AC:92:32:BF:95:4D".toString());
    BluetoothSocket mmSocket;
    BluetoothDevice mmDevice = serwer;

    BluetoothSocket tmp = null;

    try {
        UUID uuid = UUID.fromString("ebe3a525-0cef-4907-94b0-e39e1255ce43");
        tmp = serwer.createRfcommSocketToServiceRecord(uuid);
    } catch (Exception e) { }
    mmSocket = tmp;

    try {
        Log.d( tag: "INFO", msg: "Próba połączenia....");
        mmSocket.connect();
        Log.d( tag: "INFO", msg: "Połączono z serwerem!");
        BufferedReader in = new BufferedReader(new InputStreamReader(mmSocket.getInputStream()));
        String input = in.readLine();
        Log.d( tag: "INFO", msg: "Serwer mówi: "+input);

        InputConnection ic = getCurrentInputConnection();
        ic.commitText(input,  newCursorPosition: 1);

    } catch (Exception ce) {
        try {
            mmSocket.close();
        } catch (Exception cle) { }
        return;
    }
}
```

Ilustracja 5: Komunikacja Bluetooth po stronie klienta

Wnioski Spostrzeżenia i podsumowanie

Aby urządzenia mogły komunikować się przez bluetooth, urządzenia muszą być wcześniej sparowane. Cała komunikacja przypomina komunikację w pomiędzy urządzeniami sieciowymi. Problemem jest wzajemne blokowanie się metod, dlatego najlepszym rozwiązaniem byłoby oparcie metod na wątkach.